European Funds Knowledge Education Development
Wrocław University of Science and Technology
European Union European Social Fund

ZPR PWr – Zintegrowany Program Rozwoju Politechniki Wrocławskiej

Laboratory – List 11

Introduction

One of the problems in graph theory is finding a path from one node to another one. This problem can be expressed more broadly: find the path from the source vertex to all vertices. One of the most famous algorithms was invented by Dijkstra. This algorithm was presented in the lecture.

List of tasks.

1. Use an implementation of class `Document` and `Graph` from previous list. Maybe it is needed to modify the way to store information about the graph in weighted form in two dimensional array.
2. In the `Main` class use any representation of `Document` objects collection.
3. Add to class `Graph` method `DijkstraSSSP(String startVertexStr)`, which implements a Dijkstra algorithm for single-source shortest paths problem. In case you have a choice during executing algorithm, analyze vertices in a lexicographical order.

For 100 points present solutions for this list till Week 13.
For 80 points present solutions for this list till Week 14.
For 50 points present solutions for this list till Week 15.
After Week 15 the list is closed.

There is an appendix on next pages.

Appendix
The solution will be automated tested with tests from console of presented below format. The test assumes, that there are documents in a collection.

If a line is empty or starts from `'#'` sign, the line have to be ignored.
In any other case, your program should print an exclamation mark and write (copy) introduced a line and then, depending on the command follow the correct procedure / function.

If the current document is equal **`null`**, the program has to write "no current document".

There is one additional command to the commands from the previous list:

If a line has a format:
`sssp <str>`
your program has to execute the `Diksjtra(str)` method, which will return a string with presentation of the paths. For every vertex lexicographically sorted there will be one line of the format:

- For starting vertex <startVertex>
  `<startVertex>=0`
- For vertex <vertexName> to which there is no path:
  `no path to <vertexName>`
- For every other <vertexName> vertex presentation of the path in the below format ended with equal sign and value of the shortest path:
  `<startVertex>-><vertex1>->…-><vertexName>=<weightOfPath>`

A simple test for this task:
INPUT:

```
#Test for Lab11
ld x
link=y(9)
eod
ld y
eod
ld a
link=b(2)
link=c(5)
eod
ld c
link=d(5)
link=f(6)
eod
ld b
link=d(3)
link=e(4)
eod
ld e
link=d(3)
link=f(4)
link=h(2)
link=g(8)
```

```
eod
ld d
link=e(3)
link=f(1)
eod
ld h
link=e(2)
link=g(1)
eod
ld g
link=h(1)
link=f(7)
eod
ld f
link=d(1)
link=e(4)
link=g(7)
eod
sssp a
sssp d
sssp w
ha
```

OUTPUT:

```
START
!ld x
!ld y
!ld a
!ld c
!ld b
!ld e
!ld d
!ld h
!ld g
!ld f
!sssp a
a=0
a->b=2
a->c=5
a->b->d=5
a->b->e=6
a->b->d->f=6
a->b->e->h->g=9
a->b->e->h=8
no path to x
no path to y
!sssp d
no path to a
no path to b
no path to c
d=0
d->e=3
d->f=1
d->e->h->g=6
d->e->h=5
no path to x
no path to y
!sssp w
error
!ha
END OF EXECUTION
```